

Some Combinatorial Problems on Halin Graphs

M.Kavin ¹, K.Keerthana ¹, N.Sadagopan ², Sangeetha.S ¹, and R.Vinothini ¹

¹ Department of Computer Science and Engineering, College of Engineering Guindy, Chennai, India.

² Indian Institute of Information Technology, Design and Manufacturing, Kanchepuram, India.

sadagopan@iiitdm.ac.in

Abstract. Let T be a tree with no degree 2 vertices and $L(T) = \{l_1, \dots, l_r\}$, $r \geq 2$ denote the set of leaves in T . An Halin graph G is a graph obtained from T such that $V(G) = V(T)$ and $E(G) = E(T) \cup \{\{l_i, l_{i+1}\} \mid 1 \leq i \leq r-1\} \cup \{l_1, l_r\}$. In this paper, we investigate combinatorial problems such as, testing whether a given graph is Halin or not, chromatic bounds, an algorithm to color Halin graphs with the minimum number of colors. Further, we present polynomial-time algorithms for testing and coloring problems.

1 Introduction

The study of special graph classes like bipartite graphs, chordal graphs, planar graphs have attracted the researchers both from theoretical and application perspective. On the theoretical front, combinatorial problems like vertex cover, odd cycle transversal which are otherwise NP-complete in general graphs have polynomial-time algorithms restricted to chordal graphs [1]. However, problems like Hamilton path, odd cycle transversal are still NP-complete even on special graph like planar graphs [2] which calls for identifying a non-trivial subclass of planar graphs where Hamilton path (cycle) is polynomial-time solvable. One such subclass is the class of Halin graphs. Halin graph is a graph obtained from a tree with no degree two vertices by joining all leaves with a cycle [5] and by construction Halin graphs are planar. Bondy [4] showed that Hamilton cycle is polynomial-time solvable on Halin graphs. Further, Barefoot [3] improved his result and showed that every Halin graph is Hamilton-connected. Special graph class like Halin graphs play a vital role in understanding the inherent complexity of the problem. Halin graphs are a good candidate graph class between the class of trees and the class of planar graphs. The reason Halin graph is popular in the literature is due to the fact many combinatorial problems restricted to planar graphs are NP-complete [2] and on trees, they are polynomial-time solvable. This leaves open the complexity of the problems in subclass of planar graphs, for example, Halin graphs. In particular, finding a maximum leaf spanning tree in general and planar graphs are NP-complete whereas on Halin graph it is polynomial-time solvable [6].

Although, chordality testing and planarity testing have received attention in the past, the related question, namely, Halin graph testing (whether a graph is Halin or not), to the best of our knowledge has not been addressed in the literature which we address in this paper. A proper coloring of a graph is an assignment of colors to the vertices such that the adjacent vertices receive different color. It is well-known that all planar graphs can be colored with 4 colors. In this paper, we present some structural observations on Halin graphs using which we characterize 3-colorable Halin graphs and 4-colorable Halin graphs.

1.1 Graph Preliminaries

Notation and definitions are as per [7, 8]. Let $G = (V, E)$ be an undirected non weighted simple graph, where $V(G)$ is the set of vertices and $E(G) \subseteq \{\{u, v\} \mid u, v \in V(G), u \neq v\}$. For a vertex $v \in V(G)$, $N_G(v) = \{u \mid \{u, v\} \in E(G)\}$. The degree of a vertex v is the size of $N_G(v)$. A vertex v is universal if $N_G(v) = V(G) \setminus \{v\}$. A graph is acyclic if contains no cycle. A tree is a connected and an acyclic graph. Let T be a tree with no degree 2 vertices and $L(T) = \{l_1, \dots, l_r\}$ denote the set of leaves in T . An Halin graph G is a graph obtained from T such that $V(G) = V(T)$ and $E(G) = E(T) \cup \{\{l_i, l_{i+1}\} \mid 1 \leq i \leq r-1\} \cup \{l_1, l_r\}$. Informally, an Halin graph can be seen as a graph constructed from a tree rooted at a vertex with no degree

two vertices and joining all leaves with a cycle. We refer this informal description as *tree-cycle* representation of Halin graph. A proper coloring of graph is a vertex coloring in which adjacent vertices receive different color. A graph is k -colorable if it can be properly colored with k colors. Let C_n denote a cycle graph on n vertices with $V(G) = \{v_1, \dots, v_n\}$ and $E(G) = \{\{v_i, v_{i+1}\} \mid 1 \leq i \leq n-1\} \cup \{v_1, v_n\}$. A wheel graph W_{n+1} on $n+1$ vertices is constructed using C_n and a universal vertex. i.e., $V(W_{n+1}) = \{v_1, \dots, v_{n+1}\}$ such that the set $\{v_1, \dots, v_n\}$ induces a cycle and $N_G(v_{n+1}) = \{v_1, \dots, v_n\}$.

2 Halin Graph Testing

In this section, we present a polynomial-time algorithm to test whether a given graph is Halin or not.

Correctness and Run-time Analysis: If the input graph G is Halin, then there exists a tree-cycle representation of G and our algorithm looks for one such representation. Since, the tree-cycle representation can have any element of $V(G)$ as its root, we run Breadth First Search from each vertex and identify for which vertex it yields the tree-cycle representation. If for all elements of $V(G)$, we do not get to see the tree-cycle representation, then we output G is not Halin. For a graph with n vertices and m edges, BFS runs in $O(n+m)$ time. Checking whether G_v induces a cycle or not incurs $O(n)$ time as $|L(T)| = O(n)$. Therefore, the overall running time of Halin graph testing is $O(n) \times O(n+m) = O(mn)$, polynomial in the input size.

Algorithm 1 Halin Graph Testing

```

for each vertex  $v$  in  $G$  do
     $S_v = \phi$ 
    Find the Breadth First Search (BFS) Tree  $T$  starting at  $v$  and let  $L(T)$  denotes the set of leaves in  $T$ 
    for each edge  $e = \{x, y\}$  in  $E(G) \setminus E(T)$  do
         $S_v = S_v \cup \{x, y\}$ 
    end for
     $G_v$  be the graph such that  $V(G_v) = S_v$  and  $E(G_v) = E(G) \setminus E(T)$ 
    if  $S_v = L(T)$  and  $G_v$  is an induced cycle then
        Output  $G$  is Halin and exit.
    else
        Break and run BFS again from another vertex
    end if
end for
Output  $G$  is not a Halin graph.

```

3 On Chromatic Bounds of Halin Graphs

It is well known that planar graphs are 4-colorable and hence, Halin graphs are 4-colorable. In this section, we present some structural observations on Halin graphs using which we characterize 3-colorable and 4-colorable planar graphs.

Theorem 1. *Let G be an Halin Graph. G does not contain W_{n+1} , where $n = 2k + 1, k \in \mathbb{N}$ if and only if $\chi(G) = 3$.*

Proof. Necessity: Consider the tree-cycle representation of G with vertex v being the root of the tree. Since G does not contain degree two vertices by definition, we observe that there exists a triangle in G . i.e., there exists $\{l_i, l_{i+1}\}$ in $L(T)$ such that l_i and l_{i+1} have a common parent p . It is easy to see that the set $\{p, l_i, l_{i+1}\}$ induces a triangle in G . Clearly, any proper coloring of G requires three colors to color $\{p, l_i, l_{i+1}\}$. Therefore,

$\chi(G) \geq 3$. Further, we know from Theorem 2 that any wheel-free Halin graph can be colored with at most 3 colors. Therefore, $\chi(G) = 3$.

Sufficiency: We prove this by contradiction. Suppose G contains W_{n+1} , where $n = 2k + 1, k \in \mathbb{N}$. Since G contains C_{2k+1} , we need three colors to color C_{2k+1} and to color W_{n+1} , we need one more color. Therefore, G is 4-colorable, contradicting the given fact that G is 3-colorable. This completes the sufficiency and hence, the theorem follows. \square

Corollary 1. *Let G be an Halin Graph which contains W_{n+1} , where $n = 2k + 1, k \in \mathbb{N}$ as an induced subgraph. Then, $\chi(G) = 4$.*

Proof. The claim follows from Theorem 1 and the fact that planar graphs are 4-colorable. \square

We call an Halin graph not containing W_{n+1} , where $n = 2k + 1, k \in \mathbb{N}$ as induced subgraph as *wheel free Halin graph*. We now present an algorithm which produces a 3-coloring of wheel free Halin graph. Further, we show that the algorithm runs in polynomial time. We use the set $\{c_1, c_2, c_3\}$ of colors with c_1 being the color with least index. *candidate color set* is the set of colors available for a vertex u at that point of iteration and our algorithm picks the least indexed color from the candidate color set to color u . For a vertex u , $color(u)$ denotes the color assigned to u and $parent(u)$ denotes the parent of u .

Algorithm 2 3-coloring of wheel free Halin Graphs

- 1: **Input:** A wheel free Halin graph G
 - 2: **Output:** 3-coloring of G
 - 3: Consider the *tree-cycle* representation of G and let T be the tree part of tree-cycle representation rooted at the vertex v . To get T , simply remove the edges in the cycle part.
 - 4: Initially, all vertices are uncolored.
 - 5: With respect to v , using T , identify the left most node (leaf) p and the right most node (leaf) q
 - 6: Assign $color(p) = c_3$ and $color(parent(q)) = c_3$
 - 7: For the vertices in the path P_{vs} , where $s = parent(p)$, assign the colors alternately from the set $\{c_1, c_2\}$
 - 8: For the vertices in the path P_{vt} , where $t = parent(parent(q))$, assign the colors alternately from the set $\{c_1, c_2\}$
 - 9: To color the remaining vertices, perform the Depth First Search starting from v , for each uncolored vertex u , maintain the *candidate color set* which is the set of available colors from $\{c_1, c_2, c_3\}$ to color u . Note some of the elements in $N_G(u)$ may be colored because of the previous steps
 - 10: From the *candidate color set*, pick the color with the least index and assign the same to $color(u)$
-

Theorem 2. *Algorithm 2 colors the input graph with 3 colors.*

Proof. To prove our claim, we observe that as per our algorithm, when we encounter an uncolored vertex u , either all vertices in $N_G(u)$ are colored or only a subset $S \subset N_G(u)$ is colored and in either case, it is colored with at most two colors from the set $\{c_1, c_2, c_3\}$. This is true for the following reasons: during DFS, when we visit an uncolored vertex u , $parent(u)$ and one of its neighbours are colored and together they have used up at most two colors. Therefore, u can be colored with the least indexed color from the candidate color set. However, for the rightmost leaf q , all three vertices in $N_G(q)$ are already colored. The Step-6 of our algorithm ensures that they still use at most two colors. Therefore, we are left with the third color to color q . Hence, the theorem. \square

References

1. F. Gavril: Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. SIAM Journal of Computing, 1(2), 180-187, 1972.
2. M.R.Garey, D.S.Johnson: Computers and intractability: A guide to the theory of NP-completeness. W.H.Freeman and Company, 1979.

3. C. A. Barefoot: Hamiltonian connectivity of the Halin graphs, *Congressus numerantium*, 58, 93-102, 1987.
4. J. A. Bondy: Pancyclic graphs : recent results, *Infinite and Finite Sets*, Colloq. Math. Soc. Janos Bolyai, Hungary, Vol.10, 181-187, 1973.
5. R. Halin: Studies on minimally n -connected graphs, *Combinatorial mathematics and its applications*, edited by D. J. A. Welsh (Academic Press, New York), 129-136, 1971.
6. Dingjun Lou and Huiquan Zhu: A note on max-leaves spanning tree problem in Halin graphs, *Australasian Journal of Combinatorics*, Vol. 29, 95-97, 2004.
7. Douglas B. West: *Introduction to Graph Theory*, 2nd Edition, 2000.
8. M.C.Golumbic: *Algorithmic graph theory and perfect graphs*, Academic Press. (1980)